

Factorization of \mathbb{Z} -homogeneous Polynomials in the First $(q-)$ Weyl Algebra

Albert Heinle and Viktor Levandovskyy

Lehrstuhl D für Mathematik, RWTH Aachen University, Aachen, Germany

Abstract. We present algorithms to factorize homogeneous polynomials in the first q -Weyl algebra and the first Weyl algebra. By homogeneous we mean homogeneous with respect to a \mathbb{Z} -grading on the first $(q-)$ Weyl algebra which will be introduced in this article. It turns out that the factorization can be almost completely reduced to commutative univariate factorization with some additional easy combinatorial steps. We implemented the algorithms in the computer algebra system SINGULAR (Decker et al. (2012), Greuel and Pfister (2007)), which provides algorithms and an interface to deal with noncommutative polynomials (Greuel et al. (2010)). The implementation beats currently available implementations dealing with factorization in the first Weyl algebra in speed and elegance of the results. Furthermore it broadens the range of polynomials we can nowadays factorize in a feasible amount of time using a computer algebra system.

1 Introduction

Operator algebras as the q -Weyl and the Weyl algebras are interesting objects to study as one can derive properties of the solution spaces of their associated equations one wants to solve. Especially concerning the problem of finding the solution, the preconditioning step of factorizing an operator can help to reduce enormously the complexity of that problem in advance.

But usually those operator algebras are noncommutative polynomial algebras, and a factorization of an element in those algebras is neither unique in the classical sense (i.e. unique up to multiplication by a unit), nor easy to compute at all in general.

Nevertheless, a lot has been done in this field in the past. S. Tsarev has studied the form, number and the properties of the factors of a differential operator in Tsarev (1994) and Tsarev (1996), where he used and extended the work presented in Loewy (1903) and Loewy (1906).

A very general approach to noncommutative algebras and their properties, including factorization, is also done by J. Bueso et al. in (Bueso et al. (2003)). They provide several algorithms there and introduce various point of views when dealing with noncommutative polynomial algebras.

In his dissertation M. van Hoeij developed an algorithm to factorize a differential operator (van Hoeij (1996)). There were several papers following that dissertation using and extending those techniques (e.g. van Hoeij (1997) and

van Hoeij and Yuan (2010)), and nowadays this algorithm is implemented in the `DETools` package of `MAPLE` (Monagan et al. (2008)) as the standard algorithm for factorization of those operators.

From a more algebraic point of view and dealing only with strict polynomial noncommutative algebras, H. Melenk and J. Apel developed a package for the computer algebra system `REDUCE` (Melenk and Apel (1994)). That package provides tools to deal with noncommutative polynomial algebras and also contains a factorization algorithm for the supported algebras.

Those algorithms and implementations are very well written and they were able to factorize a large amount of polynomials we gave them as input. Nonetheless, as we will see in this paper, there exists a large class of polynomials that seem to form the worst case for those algorithms. There, one can use a different approach to obtain a factorization very quickly. This approach was originally developed in Heinle (2010), and further extended and presented at the conference ISSAC in Munich, 2010, the DEAM2 workshop in Linz, 2011, and the conference “Computeralgebra Tagung” in Kassel, 2012. Now, this is the first journal version of this work. Furthermore, the algorithms were implemented in `SINGULAR`, and since version 3-1-3 they became part of the distribution (library: `ncfactor.lib`).

1.1 Basic Notions and Definitions

We will start with introducing the first q -Weyl algebra and the first Weyl algebra. By \mathbb{K} we always denote an arbitrary field.

Definition 1.1. *The **first q -Weyl algebra** Q_1 is defined as*

$$Q_1 := \mathbb{K}\langle x, \partial \mid \partial x = qx\partial + 1 \rangle,$$

where q is a unit in \mathbb{K} . Q_1 is the operator algebra associated to

$$\partial_q : f(x) \mapsto \frac{f(qx) - f(x)}{(q-1)x},$$

also known as the q -derivative, where $f \in \mathbb{K}[x]$. For further reading consider Kac and Cheung (2002).

For $q = 1$, the operator is still well defined. This can be seen in the following way. Let $f = \sum_{i=0}^n a_i x^i$, where $n \in \mathbb{N}_0$ and $a_i \in \mathbb{K}$. Then

$$f(qx) - f(x) = \sum_{i=0}^n a_i (qx)^i - \sum_{i=0}^n a_i x^i = \sum_{i=0}^n a_i x^i (q^i - 1).$$

The expression $q - 1$ is clearly a divisor of $q^i - 1$ for all $i \geq 1$, and we obtain

$$\frac{f(qx) - f(x)}{(q-1)x} = \sum_{i=1}^n a_i x^{i-1} \left(\sum_{j=0}^{i-1} q^j \right).$$

For the special case where $q = 1$ we have the **first Weyl algebra**, which is denoted by A_1 .

The first (q -) Weyl algebra possesses a nontrivial \mathbb{Z} -grading – introduced by M. Kashiwara and B. Malgrange in 1983 (Kashiwara (1983), Malgrange (1983)) – using the weight vector $[-v, v]$ for a $v \in \mathbb{Z}$ on the tuple $[x, \partial]$. For simplicity, we will choose $v := 1$. In what follows, \deg denotes the degree induced by this weight vector. By homogeneous polynomials, we also mean homogeneous with respect to that weight vector.

Example 1.2. We have

$$\deg(x\partial) = \deg(\partial x + 1) = 0.$$

The polynomial

$$x\partial^2 + x^4\partial^5 + \partial$$

is also a homogeneous polynomial, but of degree one.

The n th graded part of Q_1 and analogously the n th graded part of A_1 is given by

$$Q_1^{(n)} := \left\{ \sum_{j-i=n} r_{i,j} x^i \partial^j \mid i, j \in \mathbb{N}_0, r_{i,j} \in \mathbb{K} \right\},$$

i.e. the degree of a monomial is determined by the difference of its powers in x and ∂ .

Concerning this choice of degree, the so called **Euler operator**

$$\theta := x\partial$$

will play an important role as we will see soon.

First of all, let us study some commutation rules the Euler operator has with x and ∂ . For Q_1 , in order to abbreviate the size of our formulas, we introduce the so called q -bracket.

Definition 1.3. For $n \in \mathbb{N}$, we define the q -**bracket** $[n]_q$ by

$$[n]_q := \frac{1 - q^n}{1 - q} = \sum_{i=0}^{n-1} q^i.$$

Lemma 1.4 (Compare with Saito et al. (2000)). In A_1 , the following commutation rules do hold for $n \in \mathbb{N}$:

$$\begin{aligned} \theta x^n &= x^n(\theta + n) \\ \theta \partial^n &= \partial^n(\theta - n). \end{aligned}$$

More general, in Q_1 the following commutation rules do hold for $n \in \mathbb{N}$:

$$\begin{aligned} \theta x^n &= x^n(q^n \theta + [n]_q) \\ \theta \partial^n &= \frac{\partial^n}{q} \left(\frac{\theta - 1}{q^{n-1}} - \frac{q^{-n+2} - q}{1 - q} \right). \end{aligned}$$

Those rules follow via induction on $n \in \mathbb{N}$ and we recommend to perform this induction in order to get experience with calculating in the first q -Weyl algebra.

Remark 1.5. With the help of the Lemma above one can also easily see that the so called shift algebra

$$\mathbb{K}\langle n, s | sn = (n+1)s \rangle$$

is a subalgebra of the first Weyl algebra A_1 . The embedding of a polynomial $p = \sum_{i=0}^n p_i(n)s^n$ in shift algebra, where $p_i \in \mathbb{K}[n]$, into the first Weyl algebra is done via the following map:

$$\iota : \mathbb{K}\langle n, s | sn = (n+1)s \rangle \rightarrow A_1, \quad \sum_{i=0}^n p_i(n)s^n \mapsto \sum_{i=0}^n p_i(\theta)\partial^n.$$

Therefore, the factorization techniques developed here can also be applied to the first shift algebra.

Those commutation rules can of course be extended to arbitrary polynomials in θ .

Corollary 1.6. *Consider $f(\theta) := f \in \mathbb{K}[\theta]$, $\theta := x\partial$. Then, in Q_1 , for all $n \in \mathbb{N}$ we have*

$$\begin{aligned} f(\theta)x^n &= x^n f(q^n\theta + [n]_q) \\ f(\theta)\partial^n &= \partial^n f\left(\frac{1}{q}\left(\frac{\theta-1}{q^{n-1}} - \frac{q^{-n+2}-q}{1-q}\right)\right), \end{aligned}$$

whereas in A_1 we have

$$\begin{aligned} f(\theta)x^n &= x^n f(\theta + n) \\ f(\theta)\partial^n &= \partial^n f(\theta - n), \end{aligned}$$

Those are the basic tools we need to explain our approach for factoring homogeneous polynomials in the first Weyl and the first q -Weyl algebra.

2 A new Approach for Factoring Homogeneous Polynomials in the First (q -) Weyl Algebra

The main idea of our factorization technique lies in the reduction to a commutative univariate polynomial subring of A_1 resp. Q_1 , namely $\mathbb{K}[\theta]$. It appears that this subring is quite huge in the sense of reducibility of its elements in A_1 resp. Q_1 .

2.1 Factoring homogeneous polynomials of degree zero

The following lemma shows, that we can rewrite every homogeneous polynomial of degree zero in A_1 and Q_1 as polynomial in $K[\theta]$.

Lemma 2.1 (Compare with Saito et al. (2000), Lemma 1.3.1). *In A_1 , we have the following identity for $n \in \mathbb{N}$:*

$$x^n \partial^n = \prod_{i=0}^{n-1} (\theta - i).$$

In Q_1 , one can rewrite $x^n \partial^n$ as element in $\mathbb{K}[\theta]$ and it is equal to

$$\frac{1}{q^{T_{n-1}}} \prod_{i=0}^{n-1} \left(\theta - \sum_{j=0}^{i-1} q^j \right) = \frac{1}{q^{T_{n-1}}} \prod_{i=0}^{n-1} (\theta - [i]_q),$$

where T_i denotes the i th triangular number, i.e.

$$T_i := \sum_{j=0}^i j = \frac{i(i+1)}{2}$$

for all $i \in \mathbb{N}_0$.

Therefore the factorization of homogeneous polynomials of degree zero can be done by rewriting the polynomial as element in $K[\theta]$ and apply a commutative factorization on the polynomial, which is well implemented in every computer algebra system.

Of course, this would not be a complete factorization, as there are still elements irreducible in $\mathbb{K}[\theta]$, but reducible in Q_1 resp. A_1 . An obvious example is θ itself.

But fortunately there are only two monic polynomials irreducible in $K[\theta]$, but reducible in A_1 resp. Q_1 . This is shown by the following Lemma.

Lemma 2.2. *The polynomials θ and $\theta + \frac{1}{q}$ are the only irreducible monic elements in $\mathbb{K}[\theta]$ that are reducible in Q_1 . For A_1 , the polynomials θ and $\theta + 1$ are the only irreducible monic elements in $\mathbb{K}[\theta]$ that are reducible in A_1 .*

Proof. We will only consider the proof for Q_1 , as the proof for A_1 is done in an analogue way. Let $f \in \mathbb{K}[\theta]$ be a monic polynomial. Assume that it is irreducible in $\mathbb{K}[\theta]$, but reducible in Q_1 . Let φ, ψ be elements in Q_1 with $\varphi\psi = f$. Then φ and ψ are homogeneous and $\varphi \in Q_1^{(-k)}, \psi \in Q_1^{(k)}$ for a $k \in \mathbb{Z} \setminus \{0\}$. Without loss of generality let k be positive. For k being negative we can use a similar argument.

Then

$$\begin{aligned} \varphi &= \tilde{\varphi}(\theta)x^k \\ \psi &= \tilde{\psi}(\theta)\partial^k \end{aligned}$$

for $\tilde{\varphi}, \tilde{\psi} \in \mathbb{K}[\theta]$.

Using Corollary 1.6, we have

$$f = \tilde{\varphi}(\theta)x^k\tilde{\psi}(\theta)\partial^k = \tilde{\varphi}(\theta)x^k\partial^k\tilde{\psi}\left(\frac{1}{q}\left(\frac{\theta-1}{q^{n-1}} - \frac{q^{-n+2}-q}{1-q}\right)\right).$$

As we know from Lemma 1.4 the equation

$$x^k\partial^k = \frac{1}{q^{T_{k-1}}} \prod_{i=0}^{k-1} \left(\theta - \sum_{j=0}^{i-1} q^j\right)$$

holds.

Thus, because we assumed f to be irreducible in $\mathbb{K}[\theta]$, we must have $\tilde{\varphi}, \tilde{\psi} \in \mathbb{K}$ and $k = 1$. Because f is monic, we must also have $\tilde{\varphi} = \tilde{\psi}^{-1}$.

As a result, the only possible f is $f = \theta$. If we originally would have chosen k to be negative, the only possibility for f would be $f = \theta + \frac{1}{q}$. This completes the proof.

Therefore, we have an algorithm to factor a homogeneous polynomial $p \in A_1$ (resp. $p \in Q_1$) of degree zero in $\mathbb{K}[\theta]$. It is done using the following steps.

1. Rewrite p as an element in $\mathbb{K}[\theta]$.
2. Factorize this resulting element in $\mathbb{K}[\theta]$ with commutative methods.
3. If there is θ or $\theta + 1$ among the factors, factorize it to $x \cdot \partial$ resp. $\partial \cdot x$.

As mentioned before, the factorization of a polynomial in a noncommutative ring is in general not unique up to multiplication by units or interchanges. Several factorizations can occur. Fortunately, in the case of the polynomial first (q -) Weyl algebra, there are only finitely many different factorizations possible due to Tsarev (Tsarev (1996)). In order to obtain all these different factorizations, one can apply the commutation rules for x and ∂ with θ . That these are all possible factorizations up to multiplication by units can be seen using an analogue approach as in the proof of Lemma 2.2. Let us consider an example for that.

Example 2.3. Let

$$p := x^3\partial^3 + 4x^2\partial^2 + 3x\partial \in A_1$$

and $\mathbb{K} := \mathbb{Q}$. It is homogeneous of degree zero and rewritten in $\mathbb{K}[\theta]$ it is equal to

$$\theta^3 + \theta^2 + \theta.$$

This polynomial factorizes in $\mathbb{K}[\theta]$ to $\theta \cdot (\theta^2 + \theta + 1)$, which further factorizes as θ is a factor to $x \cdot \partial \cdot (\theta^2 + \theta + 1)$. To get all possible factorizations of p , we apply the commutation rules with x resp. ∂ and obtain the following other factorizations.

$$\begin{aligned} &(\theta^2 + \theta + 1) \cdot x \cdot \partial, \\ &x \cdot (\theta^2 + 3\theta + 3) \cdot \partial. \end{aligned}$$

Remark 2.4. Let us consider the complexity of the first two steps mentioned above.

Ad step 1: The polynomial p has due to the assumption of being homogeneous of degree zero the form

$$p = \sum_{i=0}^n p_i x^i \partial^i, n \in \mathbb{N}, p_i \in \mathbb{K}.$$

In order to transform it into an element in $\mathbb{K}[\theta]$, we have to apply the rewriting rule stated in Lemma 2.1 for every term $x^i \partial^i$ in p . For that, one can make use of the fact that

$$x^{n+1} \partial^{n+1} = x^n \partial^n \cdot (\theta - n).$$

Thus, in order to perform step 1, we need to perform for every $i \in \mathbb{N}$ a multiplication of a polynomial in $\mathbb{K}[\theta]$ of degree i with a polynomial of degree 1.

Ad step 2: Unfortunately, the factorization problem even in the univariate case does not have polynomial complexity. In general we face exponential complexity. But there are algorithms developed since the 1980s that appeared to be very fast in practice. One example of it is the famous LLL algorithm by Lenstra, Lenstra, Lavász, László developed in 1982 (Lenstra et al. (1982)). For further readings on the complexity of the factorization problem we recommend Kaltofen (1982).

2.2 Factoring homogeneous polynomials of arbitrary degree

Fortunately, the hard work is already done and the factoring of homogeneous polynomials of arbitrary degree is just a small further step.

The following well known Theorem reveals why.

Theorem 2.5. $Q_1^{(0)}$ resp. $A_1^{(0)}$ is a ring, generated by the element $\theta := x\partial$ as a \mathbb{K} -algebra. The other graded direct summands $Q_1^{(k)}$ resp. $A_1^{(k)}$ are cyclic $Q_1^{(0)}$ resp. $A_1^{(0)}$ modules generated by the element x^{-k} , if $k < 0$, or by ∂^k , if $k > 0$.

Proof. The first statement can be seen using Lemma 2.1, as we can identify $Q_1^{(0)}$ resp. $A_1^{(0)}$ with $\mathbb{K}[\theta]$.

For the second statement recall that for a polynomial $p \in Q_1^{(k)}$ resp. $p \in A_1^{(k)}$ being homogeneous of degree $k \in \mathbb{Z}$ means, that the distance of the powers of x and ∂ in every monomial of p equals k resp. $-k$. That means, we can divide by x^{-k} resp. ∂^k from the right and obtain the desired module structure.

Therefore, one factorization of a homogeneous polynomial $p \in Q_1^{(k)}$ resp. $p \in A_1^{(k)}$ of degree $k \in \mathbb{Z}$ can be obtained using the following steps.

1. Divide p by x^{-k} resp. ∂^k from the right.
2. Factorize the remaining polynomial – which is homogeneous of degree zero – using the steps shown in the previous subsection.

This will lead to one factorization. Again, to obtain all possible factorizations, one simply swaps the factors homogeneous of degree zero with the ∂ s resp. x s. How this is precisely done is described in Algorithm 2.

Now we can finally state the algorithm to factorize homogeneous polynomials in the first $(q-)$ Weyl algebra.

To obtain one factorization, we simply use the Algorithm 1.

Algorithm 1 HomogFac: Factorization of a homogeneous polynomial in the first $(q-)$ Weyl algebra

Input: $h \in A_1^{(m)}$ (resp. $h \in Q_1^{(m)}$), where $m \in \mathbb{Z}$

Output: $(f_1, \dots, f_n) \in A_1^n$ resp. $(f_1, \dots, f_n) \in Q_1^n$, such that $f_1 \cdot \dots \cdot f_n = h$, $n \in \mathbb{N}$

```

1: if  $m \neq 0$  then
2:   if  $m < 0$  then
3:     Get  $\hat{h} \in A_1^{(0)}$  such that  $h = \hat{h}x^{-m}$ 
4:      $factor := (\underbrace{x, \dots, x}_{-m \text{ times}})$ 
5:   else
6:     Get  $\hat{h}$  such that  $h = \hat{h}\partial^m$ 
7:      $factor := (\underbrace{\partial, \dots, \partial}_{m \text{ times}})$ 
8:   end if
9: else
10:   $\hat{h} := h$ 
11:   $factor := 1$ 
12: end if
13:  $(\hat{f}_1, \dots, \hat{f}_l) := \text{Factorization of } \hat{h} \text{ as element in } \mathbb{K}[\theta] \text{ } (l \in \mathbb{N})$ 
14:  $(\hat{\hat{f}}_1, \dots, \hat{\hat{f}}_l) := \text{Substitute } \theta \text{ by } x \cdot \partial \text{ in } (\hat{f}_1, \dots, \hat{f}_l)$ 
15:  $result := ()$ 
16: for  $i$  from 1 to  $l$  do
17:   if  $\hat{\hat{f}}_i = x \cdot \partial$  then
18:     Append  $x$  and  $\partial$  to  $result$ 
19:   else
20:     if  $\hat{\hat{f}}_i = \partial \cdot x$  then
21:       Append  $\partial$  and  $x$  to  $result$ 
22:     else
23:       Append  $\hat{\hat{f}}_i$  to  $result$ 
24:     end if
25:   end if
26: end for
27: Append each element in  $factor$  to  $result$ 
28: return  $result$ 

```

To obtain all factorizations, we extend Algorithm 1 by Algorithm 2.

The termination of those algorithms is clear, as we always only iterate over finite sets. The correctness follows by our preliminary work.

Algorithm 2 HomogFacAll: All factorizations of a homogeneous polynomial in the first $(q-)$ Weyl algebra

Input: $h \in A_1^{(m)}$ (resp. $h \in Q_1^{(m)}$), where $m \in \mathbb{Z}$

Output: $\{(f_1, \dots, f_n) \in A_1^n \mid f_1 \cdot \dots \cdot f_n = h, n \in \mathbb{N}\}$

```

1:  $(f_1, \dots, f_\nu, g, \dots, g) := \text{HomogFac}(h)$ 
    $\{\nu \in \mathbb{N}_0, g \in \{x, \partial\}, f_i \in A_1^{(0)} \text{ concatenated, if needed}\}$ 
2: Rewrite each  $f_i$  as element in  $\mathbb{K}[\theta]$ 
3:  $result := \{\text{Permutations of } (f_1, \dots, f_\nu, g, \dots, g) \text{ with respect to the commutation rules}\}$ 
4: for  $(g_1, \dots, g_n) \in result$  do
5:   for  $i$  from 1 to  $n$  do
6:     if  $g_i = \theta$  (resp.  $g_i = \theta + 1$ ) then
7:        $g_i := x, \partial$ 
8:        $leftpart := (g_1, \dots, g_{i-1}, x)$ 
9:        $rightpart := (\partial, g_{i+1}, \dots, g_n)$ 
10:      swap  $x$  down in the first list and  $\partial$  up in the second
11:      put all possibilities of the left and the right part together and add them to  $result$ 
12:     end if
13:   end for
14: end for
15: return  $result$ 

```

Remark 2.6. We made the general assumption that we are dealing with the polynomial first Weyl algebra. That means that we assumed the variables x and ∂ as polynomial indeterminates. In practice one often finds functions that are given polynomial in ∂ , but rational in x . The factorization algorithm in MAPLE factorizes exactly those functions. They have some odd properties, as there might be infinitely many distinct factorizations. But fortunately, as we studied in Heinle (2012), one can lift factorizations of a polynomial from the rational Weyl algebra to a polynomial in the Weyl algebra as defined in this paper. Furthermore, we can even say that if an element is irreducible in the polynomial first Weyl algebra, then it is also irreducible in the rational first Weyl algebra (see Heinle (2012), Chapter 1, Lemma 3.13).

3 Implementation and benchmarking

We implemented the presented algorithms in SINGULAR, and since version 3-1-3 they are part of the distribution of SINGULAR. The following example shows how to use the library containing them.

Example 3.1. Let $h \in Q_1$ be the polynomial

$$\begin{aligned}
h := & q^{25}x^{10}\partial^{10} + q^{16}(q^4 + q^3 + q^2 + q + 1)^2x^9\partial^9 \\
& + q^9(q^{13} + 3q^{12} + 7q^{11} + 13q^{10} + 20q^9 + 26q^8 \\
& + 30q^7 + 31q^6 + 26q^5 + 20q^4 + 13q^3 + 7q^2 + 3q + 1)x^8\partial^8 \\
& + q^4(q^9 + 2q^8 + 4q^7 + 6q^6 + 7q^5 + 8q^4 + 6q^3 + 4q^2 + 2q + 1) \\
& (q^4 + q^3 + q^2 + q + 1)(q^2 + q + 1)x^7\partial^7 \\
& + q(q^2 + q + 1)(q^5 + 2q^4 + 2q^3 + 3q^2 + 2q + 1) \\
& (q^4 + q^3 + q^2 + q + 1)(q^2 + 1)(q + 1)x^6\partial^6 \\
& + (q^{10} + 5q^9 + 12q^8 + 21q^7 + 29q^6 + 33q^5 \\
& + 31q^4 + 24q^3 + 15q^2 + 7q + 12)x^5\partial^5 + 6x^3\partial^3 + 24
\end{aligned}$$

and $\mathbb{K} = \mathbb{Q}$. We can use SINGULAR to obtain all of its factorizations in the following way.

```

LIB "ncfactor.lib";
ring R = (0,q),(x,d),dp;
def r = nc_algebra (q,1);
setring(r);
poly h = ... //See the polynomial defined above.
homogfacFirstQWeyl_all(h);
[1]:
  [1]:
    1
  [2]:
    x5d5+x3d3+4
  [3]:
    x5d5+6
[2]:
  [1]:
    1
  [2]:
    x5d5+6
  [3]:
    x5d5+x3d3+4

```

As one can see here, the output is a list containing lists containing elements in Q_1 . Those elements in Q_1 are factors of h , and each list represents one possible factorization of h .

If the user is interested in just one factorization the command `homogfacFirstQWeyl` instead of `homogfacFirstQWeyl_all` can be used. The output will then be just one list containing elements in Q_1 .

On the computer we used – 2 GB RAM, 2.33GHz Dual Core processor – this calculation needs 2.8 seconds. Compared to the factorization of

$$(x^5\partial^5 + 6)(x^5\partial^5 + x^3\partial^3 + 4)$$

as element in A_1 , which takes less than a second, this seems to be way more slow considering that both algorithms have the same complexity. But this slowdown is not due to more steps that need to be done in the algorithm for the q -Weyl algebra, but due to the parameter q and the speed of calculating in $\mathbb{Q}(q)$ as the basefield instead of just in \mathbb{Q} .

In fact, there is no computer algebra system known to the authors that can factor polynomials in the first q -Weyl algebra Q_1 . Therefore, we cannot compare our algorithms in this case to other implementations.

For the first Weyl algebra A_1 , there are other implementations. We can draw a comparison to the `Dfactor` method in the `DETools` package of MAPLE and the `nc_factorize_all` method in the `NCPoly` library of REDUCE. We used version 16 of MAPLE and version 3.8 of REDUCE.

Remark 3.2. As mentioned before, the algorithm implemented in MAPLE factorizes over the rational Weyl algebra, i.e. the variable x is a rational argument having adjusted commutation rules with ∂ . This is a weaker assumption on the input since the ring that is dealt with there is larger. The comparison is still valid, as we are interested in the fact if a factorization can be found at all and – if so – whether we can lift it to a polynomial factorization without multiplying by xs .

We will not go into detail about how the algorithm in MAPLE works, as it would take several pages to describe it. The interested reader can find details in van Hoeij (1997). It works with collections of exponential parts and their multiplicities at all singularities of a given differential operator f and subsequent calculation of left and right hand factors.

The algorithm implemented in REDUCE is also working with the polynomial Weyl algebra. In fact, the algorithm written there can be applied to a large amount of polynomial noncommutative rings.

Details about the functionality of the algorithm in REDUCE are unfortunately not available. One can only try to understand it from the code that is given open source. It uses several Gröbner Basis computations in order to find its solutions.

Example 3.3. Consider again the element

$$h := (x^5 \partial^5 + 6) \cdot (x^5 \partial^5 + x^3 \partial^3 + 4).$$

- SINGULAR: Found two factorizations in less than a second.
- MAPLE: Found one factorization after 29 seconds; The factors are huge (size of the output file is around 100KB).
- REDUCE: Did not terminate after 9 hours of calculation.

Example 3.4. We experimented with other examples of homogeneous polynomials in the first Weyl algebra. The results are listed in the next table. An entry labeled with “– NT –” stands for “no termination after four hours”, and the abbreviation “fcts” stands for “factorizations”.

Singular	Maple	REDUCE
$(x^{10}\partial^{10} + 5x\partial + 7) \cdot x^2 \cdot (x^{11}\partial^{11} + 3x^7\partial^7 + x\partial + 4):$		
0.37s; 12 fcts.	- NT -	- NT -
$(x^5\partial^5 + 6) \cdot (x^5\partial^5 + x^3\partial^3 + 4) \cdot \partial^{10}:$		
4.8s; 132 fcts.	31.62s; 1 fcts.	- NT -
$(5x^{10}\partial^{10} + 7x^9\partial^9 + 8x^8\partial^8 + 9x^7\partial^7 + 6x^6\partial^6 + 5x^5\partial^5 + 8x^4\partial^4 + 5x^3\partial^3 + 9x^2\partial^2 + 9x\partial + 6) \cdot \partial^{20}:$		
0.99s; 21 fcts.	- NT -	- NT -
$(7x^{15}\partial^{15} + x^{13}\partial^{13} - x^{12}\partial^{12} - 3x^{10}\partial^{10} + 2x^9\partial^9 + x^8\partial^8 + x^7\partial^7 - x^5\partial^5 - 9x^4\partial^4 + x\partial - 1) \cdot (8x^{13}\partial^{13} + 3x^{12}\partial^{12} + x^{11}\partial^{11} - 2x^{10}\partial^{10} + 10x^8\partial^8 - 3x^7\partial^7 + 2x^5\partial^5 + x^4\partial^4 + 38x\partial + 1) \cdot \partial^6:$		
36.02s; 504 fcts.	- NT -	- NT -
$(x^{10}\partial^{10} + 23x^9\partial^9 + 3x^8\partial^8 - 9x^7\partial^7 - x^5\partial^5 + 3x^4\partial^4 + 6x^3\partial^3 + 4x\partial + 1) \cdot (-x^8\partial^8 + 4x^7\partial^7 - x^6\partial^6 + 4x^5\partial^5 - 5x^4\partial^4 + x^2\partial^2 - 7x\partial - 10) \cdot x^{10}:$		
5.63s; 132 fcts.	- NT -	- NT -
$(-2x^{24}\partial^{24} + x^{23}\partial^{23} + 4x^{22}\partial^{22} - 110x^{21}\partial^{21} + x^{20}\partial^{20} + x^{19}\partial^{19} + x^{18}\partial^{18} + x^{17}\partial^{17} + 5x^{16}\partial^{16} - 7x^{15}\partial^{15} + 4x^{14}\partial^{14} - x^{13}\partial^{13} + x^{12}\partial^{12} - 2x^{11}\partial^{11} + x^9\partial^9 + 5x^8\partial^8 + x^7\partial^7 + 6x^5\partial^5 + x^4\partial^4 + 2x^3\partial^3 + 219x^2\partial^2 + x\partial - 1) \cdot (-x^{25}\partial^{25} + x^{24}\partial^{24} - 32x^{23}\partial^{23} + x^{22}\partial^{22} + 7x^{21}\partial^{21} + 61x^{20}\partial^{20} - 2x^{18}\partial^{18} + x^{16}\partial^{16} + 2x^{15}\partial^{15} - 2x^{14}\partial^{14} - x^{12}\partial^{12} - 3x^{11}\partial^{11} + 2x^{10}\partial^{10} + 2x^8\partial^8 - 9x^7\partial^7 - x^6\partial^6 + x^5\partial^5 + 4x^3\partial^3 + x^2\partial^2):$		
125.43s; 230 fcts.	- NT -	- NT -
$(x^{10}\partial^{10} + 13x^9\partial^9 - x^8\partial^8 + 4x^7\partial^7 + 13x^6\partial^6 - 3x^5\partial^5 - 37x^4\partial^4 - x^3\partial^3 + x^2\partial^2 + x\partial - 1) \cdot (-x^{10}\partial^{10} - 23x^9\partial^9 + 3x^8\partial^8 + x^7\partial^7 - x^6\partial^6 - 2x^5\partial^5 - 2x^4\partial^4 + 2x^3\partial^3 - x^2\partial^2 - 2x\partial - 2):$		
0.27s; 6 fcts	- NT -	- NT -
$(98x^{15}\partial^{15} + 40x^{14}\partial^{14} + 98x^{13}\partial^{13} + 44x^{12}\partial^{12} + 55x^{11}\partial^{11} + 96x^{10}\partial^{10} + 95x^9\partial^9 + 7x^8\partial^8 + 56x^7\partial^7 + 56x^6\partial^6 + 40x^5\partial^5 + 11x^4\partial^4 + 40x^3\partial^3 + 78x^2\partial^2 + 13x\partial + 19) \cdot (61x^{15}\partial^{15} + 50x^{14}\partial^{14} + 83x^{13}\partial^{13} + 11x^{12}\partial^{12} + 89x^{11}\partial^{11} + 55x^{10}\partial^{10} + 81x^9\partial^9 + 63x^8\partial^8 + 22x^7\partial^7 + 10x^6\partial^6 + 35x^5\partial^5 + 90x^4\partial^4 + 60x^3\partial^3 + 20x^2\partial^2 + 30x\partial + 43):$		
0.23s; 2 fcts.	-NT -	- NT -
$(85x^{20}\partial^{20} + 80x^{19}\partial^{19} + 27x^{18}\partial^{18} + 74x^{17}\partial^{17} + 49x^{16}\partial^{16} + 95x^{15}\partial^{15} + 96x^{14}\partial^{14} + 37x^{13}\partial^{13} + 26x^{12}\partial^{12} + 93x^{11}\partial^{11} + 39x^{10}\partial^{10} + 19x^9\partial^9 + 48x^8\partial^8 + 82x^7\partial^7 + 26x^6\partial^6 + 26x^5\partial^5 + 7x^4\partial^4 + 61x^3\partial^3 + 8x^2\partial^2 + 81x\partial + 88)^2:$		
0.22s; 1 fcts.	- NT -	- NT -

The conclusion we can draw at this point is: Even if homogeneous polynomials seem to be easy objects to factorize, they seem to form a worst case class for the implementations in REDUCE and MAPLE.

Therefore, with our algorithm we are now able to factorize more polynomials using computer algebra systems: In general homogeneous polynomials in Q_1 , and for A_1 we have broadened the range of polynomials that can be factorized in a feasible amount of time or even sometimes at all.

4 Conclusion and Future Work

With this paper, we contributed an algorithm and an implementation for factorization of homogeneous polynomials in the first q -Weyl algebra.

Furthermore, we also considered the special case of the first Weyl algebra and showed that our algorithm beats for the large class of $[-1, 1]$ homogeneous polynomials current implementations in terms of speed and elegance of the solutions.

We can also construct a family of polynomials where the implementation in SINGULAR is the only one that is able to factorize those elements in a feasible amount of time and memory consumption.

The next thing to deal with would be general polynomials in the first (q -) Weyl algebra. A first attempt to that we did in Heinle (2010). We made highly use of our knowledge about the grading of the first Weyl algebra. The approach has been almost completely of combinatorial nature. Its speed and quality of solutions was comparable in many cases to the other implementations, but it was not elegant. It is also distributed with SINGULAR since version 3-1-3.

Recently, we developed inter alia some new techniques to factorize general polynomials in the first Weyl algebra in Heinle (2012). They are way more elegant and provide a good performance and complexity. The implementation of those techniques will soon be committed to SINGULAR and are distributed since version 3-1-6.

But still, there is a lot of work to do in that field and as we have seen in this paper, it is worth to consider special cases within the factorization problem as there are often way more elegant ways to deal with them. In our case, it was a very large class of polynomials.

Another interesting question would be factorization in the n th Weyl algebra and how similar approaches can be applied there.

Acknowledgements

Whom ever we want to acknowledge

References

- J. Bueso, J. Gómez-Torrecillas, and A. Verschoren. *Algorithmic methods in non-commutative algebra. Applications to quantum groups*. Dordrecht: Kluwer Academic Publishers, 2003.
- W. Decker, G.-M. Greuel, G. Pfister, and H. Schönemann. SINGULAR 3-1-6 — A computer algebra system for polynomial computations. 2012. <http://www.singular.uni-kl.de>.
- G.-M. Greuel and G. Pfister. *A Singular introduction to commutative algebra. With contributions by Olaf Bachmann, Christoph Lossen and Hans Schönemann. 2nd extended ed.* Berlin: Springer, 2007.

- G.-M. Greuel, V. Levandovskyy, A. Motsak, and H. Schönemann. PLURAL. A SINGULAR 3.1 Subsystem for Computations with Non-commutative Polynomial Algebras. Centre for Computer Algebra, TU Kaiserslautern, 2010. URL <http://www.singular.uni-kl.de>.
- A. Heinle. Factorization of polynomials in a class of noncommutative algebras. Bachelor Thesis at RWTH Aachen University, April 2010.
- A. Heinle. Factorization, similarity and matrix normal forms over certain ore domains. Master’s Thesis at RWTH Aachen University, September 2012.
- Victor Kac and Pokman Cheung. *Quantum calculus*. New York, NY: Springer, 2002.
- E. Kaltofen. *On the complexity of factoring polynomials with integer coefficients*. PhD thesis, Rensselaer Polytechnic Institute, 1982.
- M. Kashiwara. Vanishing cycle sheaves and holonomic systems of differential equations. , 1983.
- A.K. Lenstra, H.W.jun. Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982. doi: 10.1007/BF01457454.
- A. Loewy. Über reduzible lineare homogene Differentialgleichungen. *Math. Ann.*, 56:549–584, 1903. doi: 10.1007/BF01444307.
- A. Loewy. Über vollständig reduzible lineare homogene Differentialgleichungen. *Math. Ann.*, 62:89–117, 1906. doi: 10.1007/BF01448417.
- Bernard Malgrange. Polynômes de Bernstein-Sato et cohomologie evanescente. *Astérisque*, 101-102:243–267, 1983.
- H. Melenk and J. Apel. *REDUCE package NCPOLY: Computation in non-commutative polynomial ideals*. Konrad-Zuse-Zentrum Berlin (ZIB), 1994.
- Michael B. Monagan, Keith O. Geddes, K. Michael Heal, George Labahn, Stefan M. Vorkoetter, James McCarron, and Paul DeMarco. *Maple Introductory Programming Guide*. Maplesoft, Waterloo ON, Canada, 2008.
- M. Saito, B. Sturmfels, and N. Takayama. *Gröbner deformations of hypergeometric differential equations*. Berlin: Springer, 2000.
- S.P. Tsarev. Problems that appear during factorization of ordinary linear differential operators. *Program. Comput. Softw.*, 20(1):27–29, 1994.
- S.P. Tsarev. An algorithm for complete enumeration of all factorizations of a linear ordinary differential operator. New York, NY: ACM Press, 1996.
- M. van Hoeij. *Factorization of linear differential operators*. Nijmegen, 1996. URL <http://books.google.de/books?id=rEmjPgAACAAJ>.
- M. van Hoeij. Factorization of differential operators with rational functions coefficients. *J. Symb. Comput.*, 24(5):537–561, 1997. doi: 10.1006/jsco.1997.0151.
- M. van Hoeij and Q. Yuan. Finding all Bessel type solutions for linear differential equations with rational function coefficients. In *Proceedings of the 2010 International Symposium on Symbolic and Algebraic Computation*, ISSAC ’10, pages 37–44, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0150-3. doi: 10.1145/1837934.1837948. URL <http://doi.acm.org/10.1145/1837934.1837948>.